Poisson Processes

Math 365

March 29, 2022

Poisson Processes

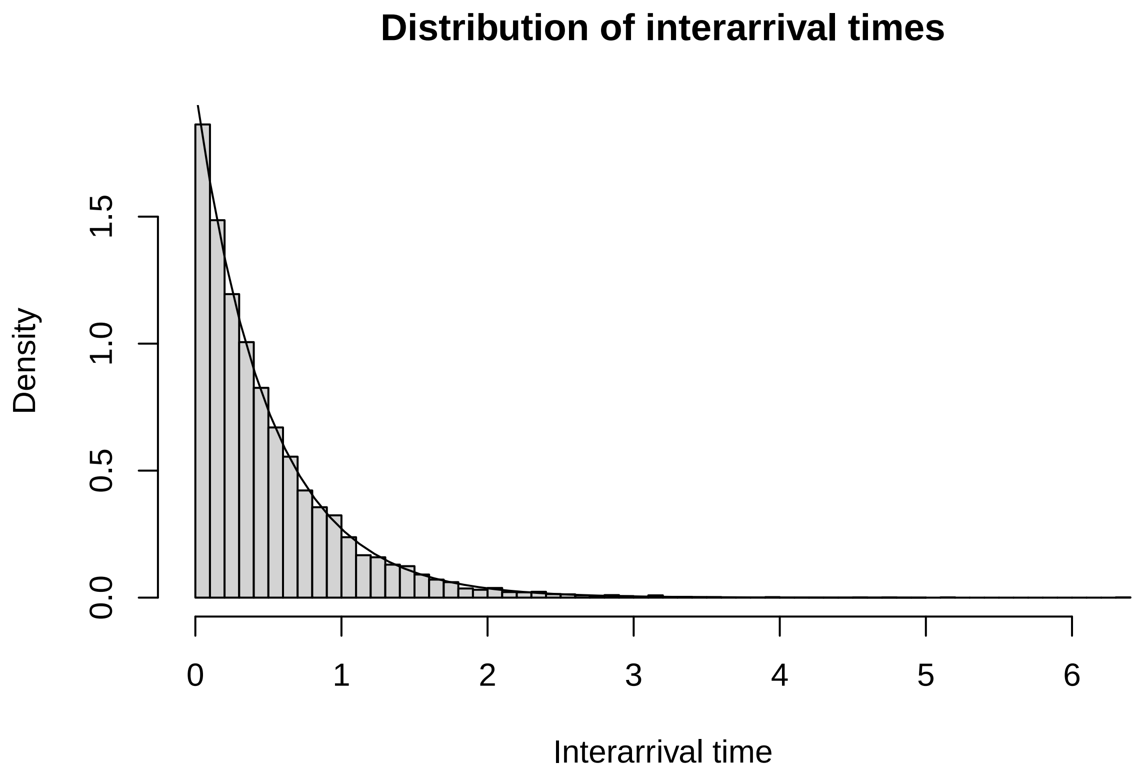
The objective of this lab is to explore basic properties and behavior of Poisson processes.

Submit the pdf knit from your completed lab to Gradescope.

Poisson process simulations

Exercise 1 Run several simulations using different values of the rate parameter λ. Visually compare the density function $f(t) = \lambda e^{-\lambda t}$ (plotted as a solid curve) to the histogram for the simulated interarrival times. You should find that the average number of arrivals per unit time during the simulation is close to the expected value λ (using the cumulative sum plot).

```
lambda <- 2 # rate parameter
totalarrivals <- 10000
X <-rexp(totalarrivals,rate=lambda) # samples interarrival times from exponential distributon
hist(X,xlab="Interarrival time",
     freq=FALSE,breaks=50,main="Distribution of interarrival times") # plots density
t <- seq(0,10/lambda,length.out=50)
points(t,lambda*exp(-lambda*t),type="l") # plots original density function on top of histogram
```

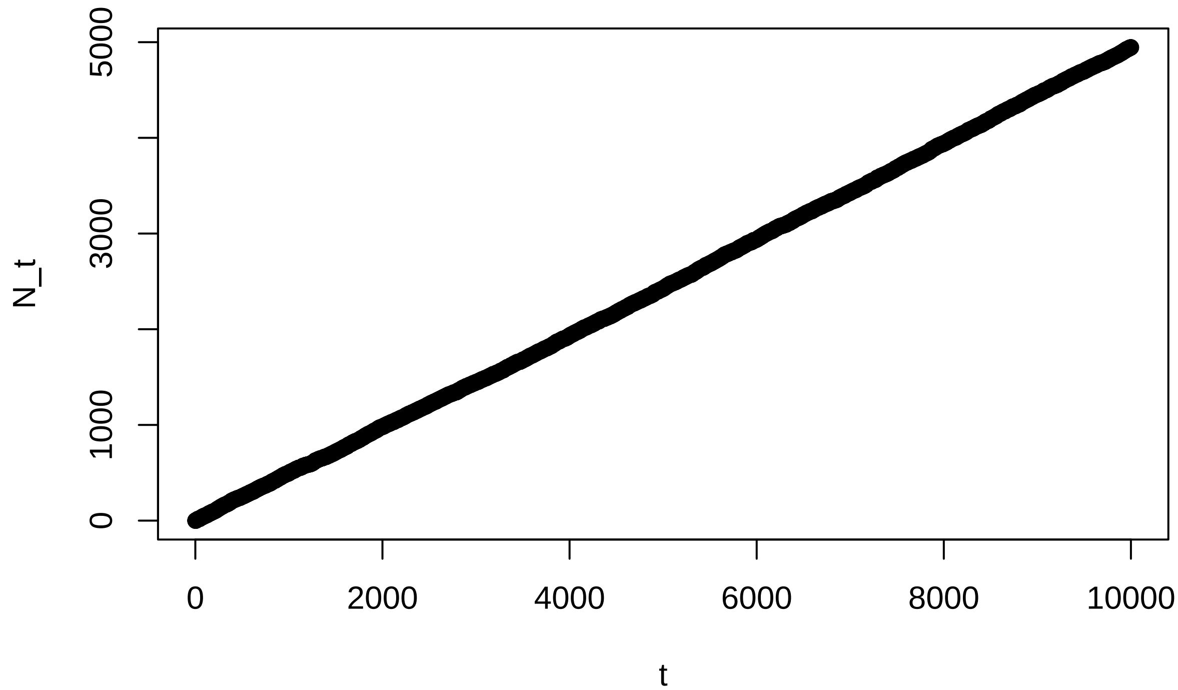



```
mean(X)
```

```
## [1] 0.4945401
```

Exercise 2 The plot showing number of arrivals N_t during time interval $[0, t]$ (with N_t on the vertical axis and time t on the horizontal axis) should look roughly like a line with what slope?

```
S <- cumsum(X)
plot(S,xlab="t",ylab="N_t") # slope will be 1/lambda
```



Real life example

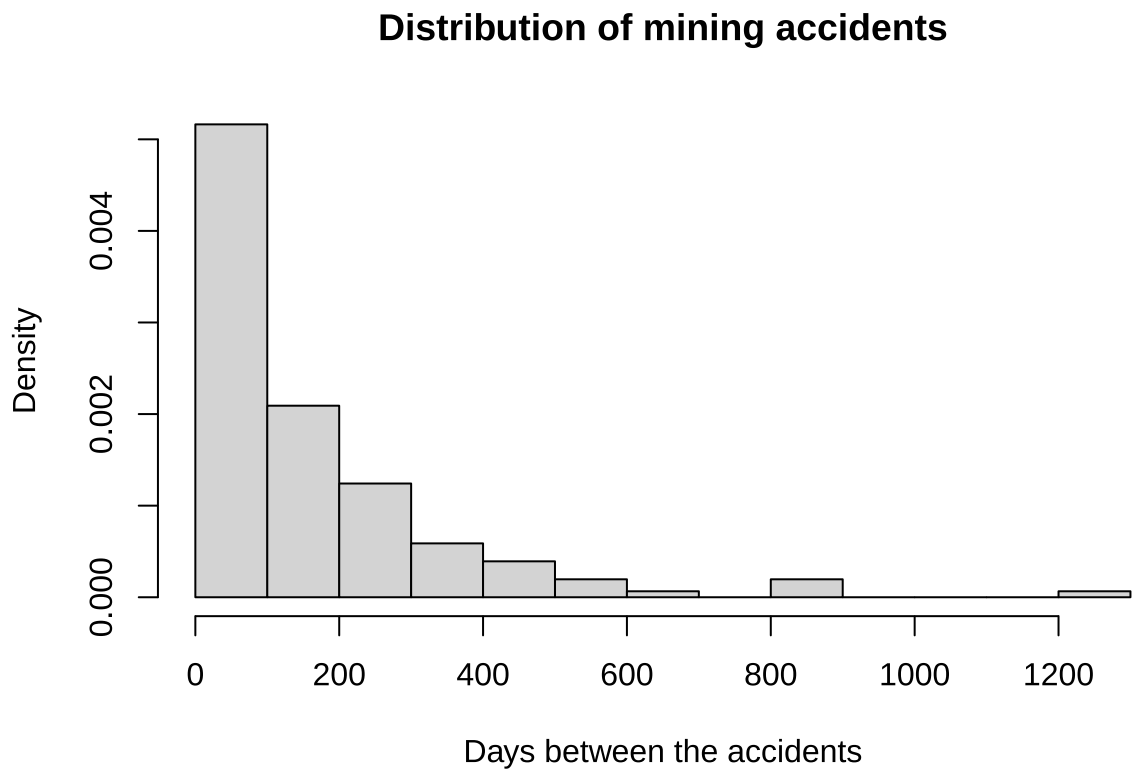
Exercise 3 Download the file CoalMineAccidents.csv from the course webpage and then load into RStudio. The setwd command sets your working directory, which needs to be the folder where you saved CoalMineAccidents.csv. The following code loads the data into RStudio as a data frame:

```
# update path to location of file on your computer
setwd("~/Desktop/SPRING 2022/stochastic processes/stochastic processes/week 8 Mar 28 - April 1/PoissonP

f <- read.table(file="CoalMineAccidents.csv", sep=",", header=TRUE)
```

This file has data on days between mining accidents in the UK. The first listed accident occurred on March 15, 1851, and the last one occurred on January 12, 1918. We can model this data as a Poisson process. Note that we will only use the Days data, not the Casualties information. We can only track time between accidents using the Poisson process, not the severity of each particular accident.

Exercise 4 Plot a histogram of the days between mining accidents to see if it looks roughly exponential:

```
hist(f[,"Days"], freq=FALSE, xlab = "Days between the accidents", main = "Distribution of mining accide
```



Exercise 5 What would you estimate for λ? A good estimate can be made by simply calculating the mean number of accidents per day from the data. Add the density function for your λ estimate to the histogram to see how well it seems to fit the data.

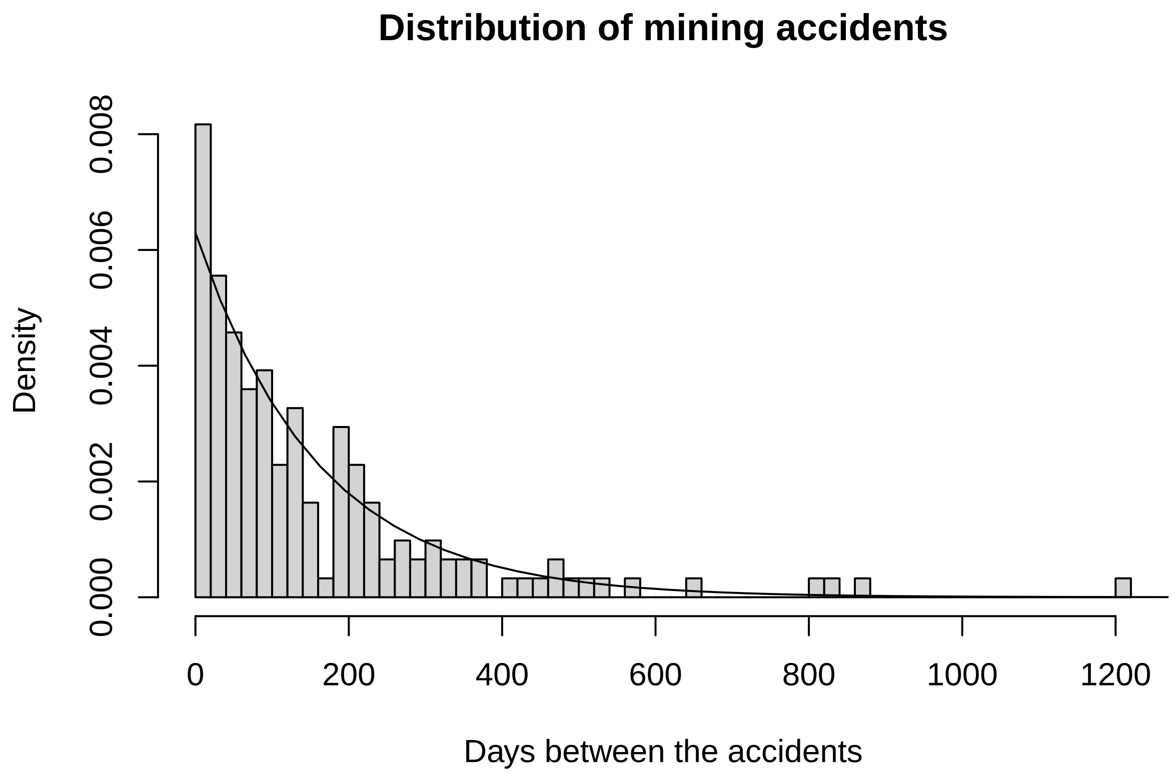
```
mean <- mean(f[,"Days"])
mean
```

```
## [1] 158.8824
```

```
lambda <- 1/mean
lambda
```

```
## [1] 0.006293965
```

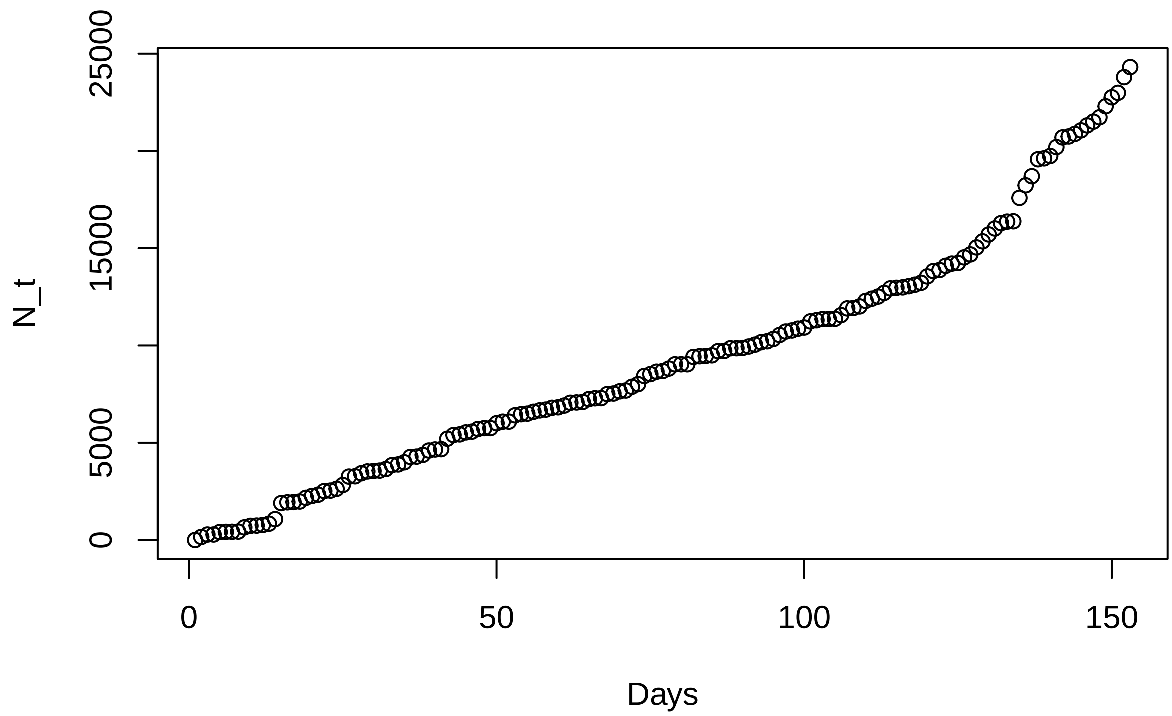
```
hist(f[,"Days"], freq=FALSE, xlab = "Days between the accidents", main = "Distribution of mining accide
t <- seq(0,10/lambda,length.out=50)
points(t,lambda*exp(-lambda*t),type="l")
```



Exercise 6 Plot the cumulative sum of the "Days" data. Does this look like a time-homogeneous process?

Ans: I think at least intially it does look like a time homogenous process.

```
S <- cumsum(f[,"Days"])
plot(S,xlab="Days",ylab="N_t") # slope will be 1/lambda
```



Exercise 7 To improve the rate parameter estimate, break the data into two time periods, splitting at the point where the rate parameter appears to have significantly changed. Estimate λ for each time period. Below is how to specify a range like the first 100 accidents:

```
l1<-1/mean(f[1:130,"Days"])
l2<-1/mean(f[130:150,"Days"])

cumsum(f[1:130,"Days"]) #TAKE THE LAST ENTRY
```

```
##   [1]     0   157   280   282   406   418   422   432   648   728   740   773
##  [13]   839  1071  1897  1937  1949  1978  2168  2265  2330  2516  2539  2631
##  [25]  2828  3259  3275  3429  3524  3549  3568  3646  3848  3884  3994  4270
##  [37]  4286  4374  4599  4652  4669  5207  5394  5428  5529  5570  5709  5751
##  [49]  5752  6002  6082  6085  6409  6465  6496  6592  6662  6703  6796  6820
##  [61]  6911  7054  7070  7097  7241  7286  7292  7500  7529  7641  7684  7877
##  [73]  8011  8431  8526  8651  8685  8812  9030  9032  9032  9410  9446  9461
##  [85]  9492  9707  9718  9855  9859  9874  9946 10042 10166 10216 10336 10539
##  [97] 10715 10770 10863 10922 11237 11296 11357 11358 11371 11560 11905 11925
## [109] 12006 12292 12406 12514 12702 12935 12963 12985 13046 13124 13223 13549
## [121] 13824 13878 14095 14208 14240 14528 14679 15040 15352 15706
```

```
15706/365
```

```
## [1] 43.03014
```

Using Gibbs sampler to determine the change point

This MCMC example was written by Joshua French to demonstrate how to find the time point at which the rate parameter changed (http://gallery.rcpp.org/articles/bayesian-time-series-changepoint/).

```
# Function for Gibbs sampler.  Takes the number of simulations desired,
# the vector of observed data, the values of the hyperparameters, the
# possible values of k, a starting value for phi, and a starting
# value for k.

# Function takes:
# nsim: number of cycles to run
# y: vector of observed values
# a, b: parameters for prior distribution of lambda
# c, d: parameters for prior distribution of phi
# kposs: possible values of k
# phi, k: starting values of chain for phi and k

gibbsloop <- function(nsim, y, a, b, c, d, kposs, phi, k)
{
  # matrix to store simulated values from each cycle
  out <- matrix(NA, nrow = nsim, ncol = 3)

  # number of observations
  n <- length(y)

  for(i in 1:nsim)
  {
    # Generate value from full conditional of phi based on
    # current values of other parameters
    lambda <- rgamma(1, a + sum(y[1:k]), b + k)

    # Generate value from full conditional of phi based on
    # current values of other parameters
    phi <- rgamma(1, c + sum(y[min((k+1),n):n]), d + n - k)

    # generate value of k
    # determine probability masses for full conditional of k
    # based on current parameters values
    pmf <- kprobloop(kposs, y, lambda, phi)
    k <- sample(x = kposs, size = 1, prob = pmf)

    out[i, ] <- c(lambda, phi, k)
  }
  out
}

# Given a vector of values x, the logsumexp function calculates
# log(sum(exp(x))), but in a "smart" way that helps avoid
# numerical underflow or overflow problems.

logsumexp <- function(x)
{
  log(sum(exp(x - max(x)))) + max(x)
```

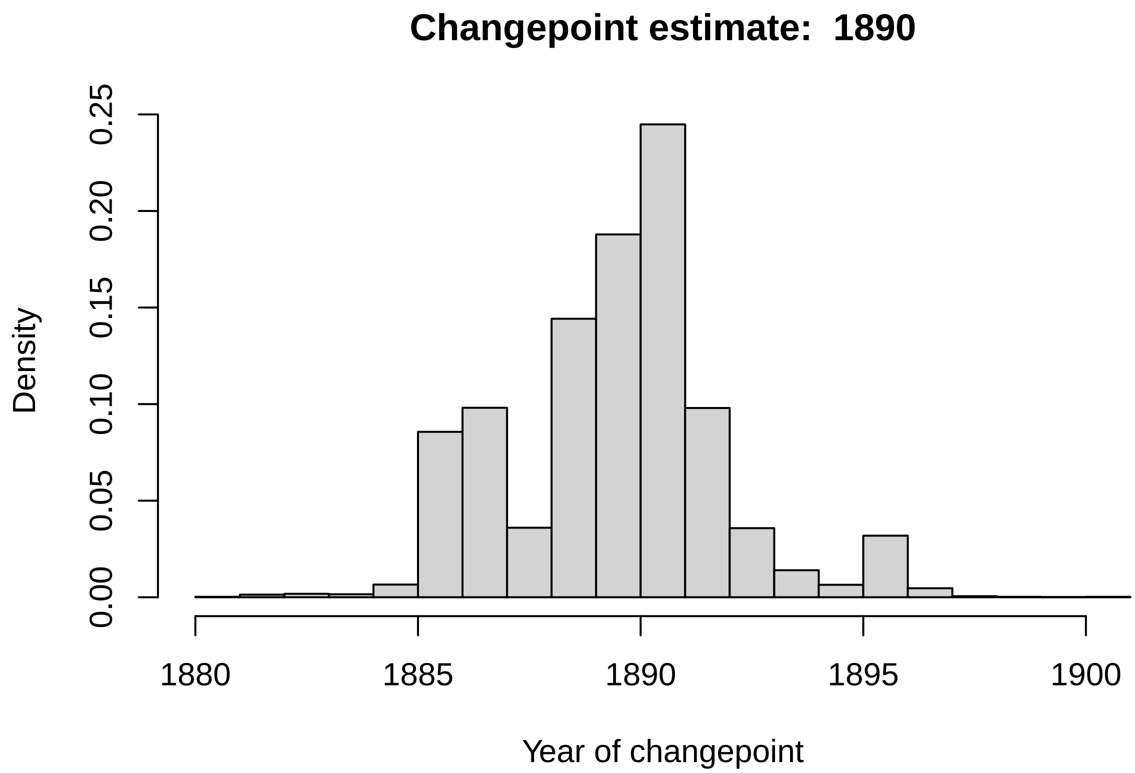
```
}

# Determine pmf for full conditional of k based on current values of other
# variables. Takes possible values of k, observed data y, current values of
# lambda, and phi. It does this naively using a loop.

kprobloop <- function(kposs, y, lambda, phi)
{
  # create vector to store argument of exponential function of
  # unnormalized pmf, then calculate them
  x <- numeric(length(kposs))
  for(i in kposs)
  {
    x[i] <- i*(phi - lambda) + sum(y[1:i]) * log(lambda/phi)
  }
  #return full conditional pmf of k
  return(exp(x - logsumexp(x)))
}
```

For this purpose, we'll use the data organized in a different way: the number of accidents each year, from 1851 to 1962.

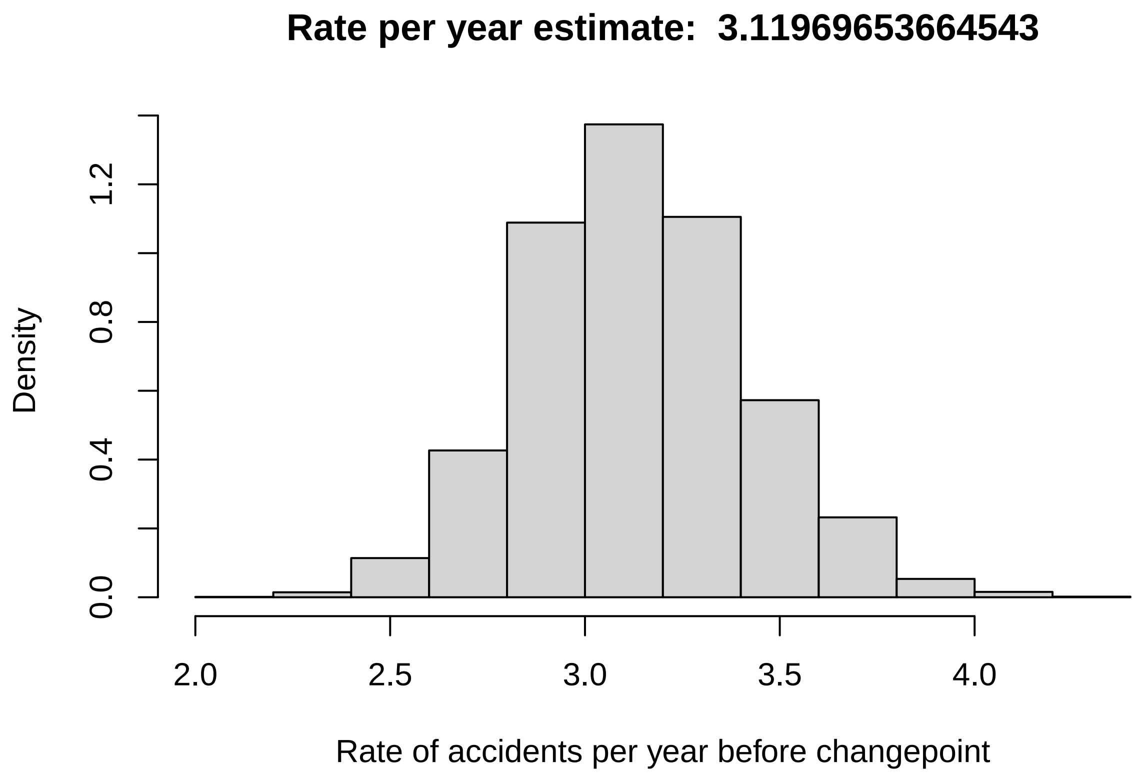
```
yr <- 1851:1962
annualaccidents <- c(4,5,4,1,0,4,3,4,0,6,3,3,4,0,2,6,3,3,5,4,5,3,1,4,4,1,5,5,3,4,2,
            5,2,2,3,4,2,1,3,2,2,1,1,1,1,3,0,0,1,0,1,1,0,0,3,1,0,3,2,2,0,1,
            1,1,0,1,0,1,0,0,0,2,1,0,0,0,1,1,0,2,3,3,1,1,2,1,1,1,1,2,4,2,0,
            0,0,1,4,0,0,0,1,0,0,0,0,0,1,0,0,1,0,1)
```

We'll use 10,000 MCMC steps to locate the change point:

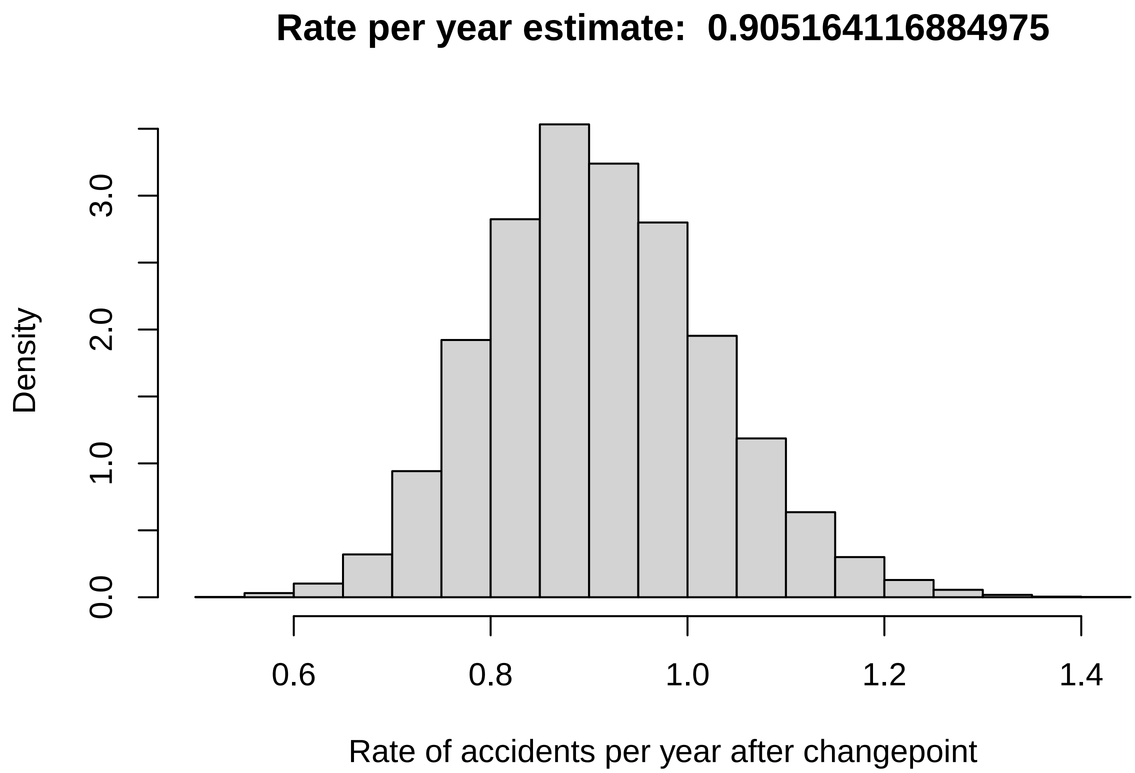
```
nsim = 10000
chain <- gibbsloop(nsim = nsim, y = annualaccidents, a = 4, b = 1, c = 1, d = 2, kposs = 1:112, phi = 1
hist(chain[1000:nsim,3]+1850,xlab="Year of changepoint",freq=FALSE,
     main=paste("Changepoint estimate: ",median(1850+chain[1000:nsim,3])))
```




```
hist(chain[1000:nsim,1],xlab="Rate of accidents per year before changepoint",freq=FALSE,
     main=paste("Rate per year estimate: ",median(chain[1000:nsim,1])))
```




```
hist(chain[1000:nsim,2],xlab="Rate of accidents per year after changepoint",freq=FALSE,
     main=paste("Rate per year estimate: ",median(chain[1000:nsim,2])))
```



Exercise 8 Compare your estimates of the breakpoint and arrival rates with the ones generated by the Gibbs sampler.

Ans: THE ESTIMATE WITH GIBBS SAMPLER WAS 1890-1851 = 39, BUT I GOT 43.3014 WHICH IS CLOSE ENOUGH ESTIMATE.